# 基于 iForest 和 LOF 的流量异常检测 *

杭菲璐 [1]，郭　威 [1]，陈何雄 [1]，张振红 [1]，易东阳 [2]

(1. 云南电网有限责任公司 信息中心，昆明 650034; 2. 电子科技大学 网络与数据安全四川省重点实验室，成都 610054)

**摘　要：**异常检测在现代大规模分布式系统的安全管理中起着重要作用，而网络流量异常检测则是组成异常检测系统的重要工具。网络流量异常检测的目的是找到和大多数流量数据不同的流量，并将这些离群点视为异常。由于现有的基于树分离的孤立森林(iForest)检测方法存在不能检测出局部异常的缺陷，为了克服这个缺陷，提出一种基于 iForest 和局部离群因子(LOF)近邻集成的无监督的流量异常检测方法。首先，改进原始的 iForest 与 LOF 算法，在提升检测精度的同时控制算法时间；然后，分别使用两种改进算法进行检测，并将结果进行融合以得到最终的检测结果；最后，在自制数据集上对所提方法进行有效性验证。实验结果表明，所提方法能够有效地隔离出异常，获得良好的流量异常检测效果。

**关键词：**流量异常检测；大规模多维数据；孤立森林；特征离群系数；局部离群因子

**中图分类号：**TP309　　doi: 10.19734/j.issn.1001-3695.2022.03.0121

# Network traffic anomaly detection based on iforest and lof

Hang Feilu[1], Guo Wei[1], Chen Hexiong[1], Zhang Zhenhong[1], Yi Dongyang[2]

(1. *Information Center*, *Yunnan Power Grid Company Limited*, *Kunming* 650034, *China*; 2. *Network & Data Security Key Laboratory of Sichuan Province*, *University of Electronic Science & Technology of China*, *Chengdu* 610054, *China*)

**Abstract:** Absrtact: Anomaly detection plays an important role in the security management of modern large-scale distributed systems. Network traffic anomaly detection is an important tool of anomaly detection system. The purpose of network traffic anomaly detection is to find the data different from most data in the traffic log, and treat these outliers as exceptions. The existing Isolation Forest (iForest) method based on tree separation has a defect: it cannot detect local anomalies. In order to overcome the defect, this paper proposes an unsupervised traffic anomaly detection method based on iForest and local outlier factor (LOF) nearest neighbor integration. Firstly, it improves the original iForest and LOF algorithms to enhance the detection accuracy and control the algorithm time; Then, it uses the two improved algorithms to detect, and fuses the results of two algorithms to get the final detection result; Finally, the method is validated on the self-made data set. Experimental results show that the method can effectively isolate anomalies, and obtain good traffic anomaly detection effect.

**Key words:** traffic anomaly detection; large scale multidimensional data; isolation forest; characteristic outlier coefficient; local outlier factor

## 0　引言

网络异常流量检测通过对流量的检测分析判断，可以尽早地发现网络中是否有入侵行为，为网络安全管理提供依据，因此，异常流量检测逐渐成为网络安全领域的研究重点[1]。网络流量异常检测是网络安全领域中重要的一部分，传统上通常使用关键字搜索或规则匹配等方法手动检查流量日志。然而日志的复杂性以及数据量的增大使得人工检测难以进行。因此提出了许多基于流量日志的异常检测方法。网络流量异常检测的目的是找到流量日志中和大多数数据不同的数据，并将这些离群点其视为异常。常用的流量异常检测算法一般分为以下几类。

a)使用基于统计学的方法。这种方法一般会构建一个概率分布模型，并通过该模型计算对象的概率，把具有低概率的对象视为异常点。

b)使用基于聚类的方法。大部分聚类算法基于数据特征的分布将数据聚集成簇，同样也被用于单维或多维数据的异常检测。但由于聚类方法不是专门用于异常检测，所以检测效果不明显。

c)使用基于密度的方法。基于密度的方法如局部离群因子算法 LOF，通过计算一个数值 score 来反映一个样本的异常程度。LOF 算法能有效避免数据密度分布不同对检测带来的影响。但 LOF 算法在面对高维数据时效果会有所下降。因为正常的数据点可能没有足够的邻居或者异常点有很多的邻居，这样计算复杂度以及定义数据之间的距离有时会很困难。

d)使用专门的异常点检测算法。对于聚类算法来说，主要任务还是将数据聚集成不同的簇，检测异常点只是一个附带的结果，而 iForest 算法的目的就是专门检测异常点。不同于之前的算法先寻找正常的数据范围，然后将不在正常区域的点视为异常点。iForest 明确地隔离异常值，是一种非常有效的异常检测算法。它适用于高维数据，但是它分割的过程是随机的，这会导致异常检测的结果不稳定。iForest 的主要优点是它的线性执行时间，这使得它与其他方法相比非常有效，因此对于

**收稿日期：**2022-03-14；**修回日期：**2022-05-07　**基金项目：**国家自然科学基金项目(62072074，62076054，62027827，61902054，62002047)；国家重点研发计划前沿科技创新专项(2019QY1405)；四川省科技创新基地(平台)和人才计划项目(2020JDJQ0020)；四川省科技支撑计划项目(2020YFSY0010)

**作者简介：**杭菲璐(1984-)，男，云南昭通人，工程师，硕士研究生，主要研究方向为网络安全攻防技术(16182038@qq.com)；郭威(1986-)，男，云南昆明人，工程师，学士，主要研究方向为网络与网络安全维护与管理；陈何雄(1984-)，男，云南曲靖人，工程师，硕士研究生，主要研究方向为网络技术及网络安全运维；张振红(1989-)，男，云南曲靖人，工程师，硕士研究生，主要研究方向为网络安全及信息系统运维；易东阳(1999-)，男，四川乐山人，学士，主要研究方向为网络安全技术.

大型数据集的挑战是一个非常有吸引力的选择。iForest 在文献[2]中显示出在高维数据集中检测全局异常的能力，并且检测效果可以与 LOF[3]和 ORCA[4]等最先进的方法相竞争。

本文结合 iForest 和 LOF 的特点，提出了一种 iForest 和 LOF 融合的改进算法来进行流量异常检测。首先，本文采用改进的 iForest 算法和改进的 LOF 算法对原始数据进行检测；然后，对两种算法的结果进行融合。与现有的基于最近邻的异常检测器(如 LOF)不同，该算法可以高效地获取离群值。文章的其余部分组织如下：第一部分描述了相关的异常检测算法。第二部分定义了一种方法来识别异常区，将局部异常与全局异常区分开来，并详细介绍了所提出的基于 iForest 和 LOF 的流量异常检测方法。第三部分提供了实验结果与评估，表明所提方法可以有效地处理大型数据集。最后，本文在第四部分进行了总结。

## 1　相关工作

### 1.1　iForest 算法

iForest 是一个独特的异常检测器，因为它使用一种隔离机制来检测异常。在孤立森林中，异常被定义为“容易被孤立的离群点”，可以将其理解为分布稀疏且离密度高的群体较远的点。在特征空间里，分布稀疏的区域表示事件发生在该区域的概率很低，因而可以认为落在这些区域里的数据是异常的。孤立森林是一种适用于连续数据的无监督异常检测方法，即不需要有标记的样本来训练，但特征需要是连续的。该算法的核心是由若干树 *iTree* 组成的森林 *Forest*。*iTree* 是一棵随机二叉树，为了构建 *iTree*，需要通过随机抽样的方式从数据集 $D$ 中选取 $n$ 个样本构成一个数据子集 $D_1$，然后从 $D_1=\{d_1,d_2,\ldots\ldots,d_n\}$ 中随机选取一个属性 $X$ 和一个分离值 $p$；最后，根据属性 $X$ 的值对每个数据 $d_i$ 进行划分；如果 $d_i(X)<p$，则将数据放在左子节点，反之，将数据放在右子节点。在这种模式下，一个 *iTree* 将被迭代构成，直到满足以下任一条件: a) 树达到了限制的高度；b)节点上只有一个样本；c) 节点上的样本所有特征都相同。在这种随机分割的策略下，异常点通常具有较短的路径。iForest 具有时间复杂度低、精度高、适应高维数据的优点，但容易丢失数据局部密度信息，因此对全局离群点的检测优势较大，不擅长处理局部离群点。由于 *iTree* 的结构是随机的，单一图形的结果不可靠，但通过大量的 *iTree*，该算法的鲁棒性得到了较大的提高。

王燕晋等人[5]通过构建孤立森林算法和二叉树模型，加强了信息数据挖掘、检测、识别过程的运算精准度，但该方法适用性还有待提高。肖峰[6]提出基于孤立森林算法的计算机网络潜在攻击检测方法。对提取的计算机网络潜在攻击谱特征进行聚类分析，结合孤立森林学习算法实现攻击检测，但算法检测时效性还有待提高。徐迪等人[7]针对配电网线损异常检测问题提出了一种基于聚类和孤立森林算法的检测方法，但本方法预设了线损异常的概率已知，概率未知时的线损异常判定问题需要进一步研究。杨晓晖等人[8]提出了基于随机超平面的隔离机制和多粒度扫描机制，使改进的孤立森林算法对复杂异常数据模式有更好的稳健性。但由于未考虑到关联属性特性，增加了算法的不确定性。李倩等人[9]提出了一种基于模糊孤立森林算法的数据异常检测方法，有效地解决了样本数据对于每一属性的异常程度不同的问题，但增加了时间开销。赵嫚等人[10]提出了一种基于模糊聚类和孤立森林的异常检测方法，适用于实际数据集中异常点较少的用电数据异常检测，但对于异常点较多的数据适用效果较差。李新鹏等人[11]提出一种根据异常偏差率大小筛选子森林异常检测器的更新策略，解决因模型随机更新导致异常检测器整体性能下降的问题，但该方法的综合性能及适用性还有待提高。

### 1.2　LOF 算法

LOF 通过计算一个数值 score 来反映一个样本的异常程度。这个数值的计算方法是：一个样本点周围的样本点所处位置的平均密度比上该样本点所在位置的密度。比值越大于 1，则该点所在位置的密度越小于其周围样本所在位置的密度，这个点就越有可能是异常点。局部离群因子算法能有效避免数据密度分布不同对检测带来的影响，但由于计算 LOF 值时需要查找每个数据点的可达距离，导致检测成本非常高，难以满足对高维数据进行高效检测的需求。

王巨灏等人[12]针对目前台区线损异常存在的判定问题，提出了基于 KNN 和 LOF 算法的台区线损异常检测方法，所提出的方法具有良好的检测性能。但该方法没有与其他算法进行比较，说服性还有待提高。刘芳等人[13]提出了快速的 Top-n 局部离群点检测算法(MTLOF)，但没有提升算法的准确率。曾冬洲等人[14]应用主成分分析法和 LOF 相结合的方法设计了变压器异常检测模型，可以实现动态实时数据的异常检测。但该方法没有与其他算法进行比较，说服性还有待提高。司方远等人[15]提出一种基于 AP-LOF 离群组检测的配电网连接验证方法，避免了判定阈值对检测结果的影响，能够准确有效地对台区内的离群组用户进行校验，提高了配电网连接验证效率。但该方法只与 LOF 算法进行实验比较，说服性还有待提高。Xu 等人[16]提出了一种优化方法，即联合调整 LOF 超参数 $k$ 进行离群点检测的启发式策略。但该方法不能保证调整后的参数将使精确度最高。仇开等人[17]提出一种加权 LOF 结合上下文判断的云环境中服务运行数据异常检测方法，能够有效检测出云环境中的服务运行数据异常。但是，该方法在的真正率方面没有提高。贺寰烨等人[18]提出了一种基于密度空间的局部离群因子算法 LOFBDS，有良好的检测效果。但在计算和时间消耗上还存在优化空间。

## 2　基于 iForest 和 LOF 的流量异常检测方法

### 2.1　数据预处理

本方法把捕获的流量数据包保存为 pcap 格式的本地文件。首先对截取的原始 pcap 包进行预处理，经过流处理程序后，pcap 文件中的数据包(packet)以流(flow)为单位进行归并，并提取流的若干特征存成 csv 格式的流量日志。该流量日志中包含的流特征如下：每条流的用户 IP、连接时间、持续时间、协议类型、出流量大小、出流量峰值、出流量均值、入流量大小、入流量峰值、入流量均值、出包大小、出包峰值、出包均值、入包大小、入包峰值、入包均值等。

此外，原始数据中存在一些列的属性为字符串例如 TCP、IP 等，这些字符串类型的数据是没办法进行训练的，需要通过一个详细的对应表将这些字符串类型的属性转换为离散的数字并进行归一化。图 1 为数据预处理流程图。

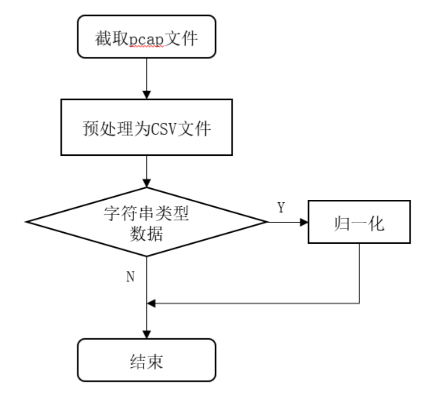



图 1　数据预处理流程图

Fig. 1　Flow chart of data preprocessing

## 2.2 改进的 iForest 算法

iForest 算法复杂度较低，但该算法仅对全局稀疏性敏感，不善于处理局部相对稀疏点。由于人为引入了随机因素，算法存在精度低和稳定性差等问题。iForest 不适用于特别高维的数据。由于每次切数据空间都是随机选取一个维度，建完树后仍然有大量的维度信息没有被使用，导致算法可靠性降低。高维空间还可能存在大量噪声维度或无关维度，影响树的构建。针对算法的特点，本文对 iForest 算法进行了改进。改进 iForest 算法的具体步骤如下：

a)随机选择数据子集。和原始的 iForest 算法一样，通过随机抽样的方式从数据集 $D$ 中选取 $n$ 个样本构成若干个数据子集 $D_i=\{d_1,d_2,......,d_n\}$，放入树的根节点。

b)通过分割建立 $iTree$。不断地分割数据子集 $D_i$ 最终形成一颗 $iTree$，再由多棵 $iTree$ 组成森林 $Forest$。原始的 iForest 算法在构建 $iTree$ 的过程中每次分割由随机选取的一个属性 $X$ 和一个分离值 $p$ 来决定，然而当异常需要通过同时考虑多个属性来检测时，仅仅选取个别属性来分离异常效果很差。因此，本文引入与原属性非轴平行的随机超平面来进行分割。在每个节点中，给定足够的随机生成超平面的实验，最终能够产生一个足够好的超平面。尽管单个超平面可能不是最优的，但由于集成学习器的聚集能力，所得模型作为一个整体仍然是高效的。具体实现为，在构造树的每一次分割中，利用随机生成的超平面构造一个分离超平面 $f$。$f$ 的表述如下:

$$f(\mathrm{X})=\sum_{j\in Q}c_j\frac{\mathrm{X}_j}{\sigma(X_j)} \tag{1}$$

其中 $Q$ 有 $q$ 个属性指标，从 $\{1,2,...,d\}$ 中随机选取而不替换；$c_j$ 是一个系数，在 $[-1,1]$ 之间随机选取；$X_j$ 是 $X$ 的第 $j$ 个属性值，$\sigma(.)$ 计算标准差。这个树构建过程继续递归地处理过滤后的子集，直到子集的大小小于或等于 2。构建多棵 $iTree$ 后，组成森林 $Forest$。

c)判断异常数据点。iForest 算法由叶子节点到树根节点的距离来计算异常值，判断异常点。在 iForest 中，样本点 $a$ 的路径长度 $h(a)$ 为从孤立树的根节点到叶子节点所经过的边的数量。给定一个包含 $n$ 个样本的数据集，树的平均路径长度为

$$c(n)=2H(n-1)-\frac{2(n-1)}{n} \tag{2}$$

其中 $c(n)$ 为给定样本数 $n$ 时，路径长度的平均值。$H(n-1)$ 的计算公式如下。

$$H(n-1)=\ln(n-1)+\xi \tag{3}$$

其中 $\xi$ 为欧拉常数，约为 0.5772。样本 $a$ 的 iForest 异常得分定义为

$$s(a,n)=2^{-\frac{E(h(a))}{c(n)}} \tag{4}$$

其中，$h(a)$ 表示样本点 $a$ 从孤立树的根节点到叶子节点所经过的边的数量，$E(h(a))$ 是样本 $a$ 在一批孤立树中 $h(a)$ 的平均值。从式(2)中可以看出，当 $E(h(a))\to c(n)$ 时，$s\to 0.5$，即当所有点的得分 $s$ 都在 0.5 左右时，样本中没有明显的异常值。当 $E(h(a))\to 0$ 时，$s\to 1$，该对象是一个异常值。当 $E(h(a))\to n-1$ 时，$s\to 0$，该对象是正常值。

改进的 iForest 使用一个简单而有效的机制，在构造 $iTree$ 的过程中引入了随机超平面，当异常检测依赖于多个属性时，在超平面中使用更多的属性可以获得更好的检测性能。

## 2.3 改进的 LOF 算法

LOF 算法的具体步骤如下：

a)对于每一个数据点，计算它与其他数据点间的欧式距离 $dist(d_i,d_j)$；

b)对于每一个数据点，计算 $k$ 距离 $k_dist(d_i)$：将数据点 $d_i$ 到其他数据点的距离按从小到大排序，数据点到第 $k$ 个数据点的距离即为 $k$ 距离。

c)对于每一个数据点，找到它的 $k$ 距离邻域 $N_k(d_i)$：到数据点 $d_i$ 的距离小于 $k_dist(d_i)$ 的数据点集合。

d)计算第 $k$ 可达距离 $reaach_dist_k(d_r,d_i)$：数据点 $d_i$ 的 $k$ 距离和数据点 $d_i$ 到数据点 $d_r$ 之间距离的最大值。

$$\begin{aligned}&reach_dist_k(d_r,d_i)\\&=\max\{K_dist(d_i),dist(d_r,d_i)\}\end{aligned} \tag{5}$$

e)计算局部可达密度 $lrd_k(d_i)$：数据点 $d_i$ 的 $k$ 距离邻域内的所有数据点到数据点 $d_i$ 的平均可达距离的倒数。

$$\begin{aligned}&lrd_k(d_i)=\\&1\bigg/\left(\frac{\sum_{d_s\in N_k(d_i)}reach_dist_k(d_s,d_i)}{|N_k(d_i)|}\right)\end{aligned} \tag{6}$$

f)计算局部离群因子 $LOF(d_i)$：数据点 $d_i$ 的 $k$ 距离邻域中所有点的局部可达密度与数据点 $d_i$ 的局部可达密度之比的平均数。

$$LOF(d_i)=\frac{\sum_{d_t\in N_k(d_i)}\frac{lrd_k(d_t)}{lrd_k(d_i)}}{|N_k(d_i)|} \tag{7}$$

g)根据局部离群因子，判断异常点。最终得到的 $LOF_k(d_i)$ 既是数据点 $d_i$ 的异常值。

使用 LOF 算法求出对象的 LOF 值，通过判断 LOF 是否近似于 1 来确定离群度。如果 LOF 远远大于 1，则认为是离群值；反之，如果接近于 1，则认为是正常点。LOF 值的计算可以根据定义得到，但随着实例数量的增加，计算成本也不断增加，到达普通用户难以接受的程度。因此，本文改进 LOF 算法，首先通过聚类来排除部分正常点，然后在剩下的集合中计算 LOF。在保证正确率的前提下大大节省了时间，具体过程如下。

首先，随机选取 $n$ 个点做为初始聚集的簇心 $a=a_1,a_2,...,a_n$，分别计算每个样本点到 $n$ 个簇核心的距离，找到离该点最近的簇核心，将它归属到对应的簇，所有点都归属到簇之后，数据集就分为了 $n$ 个簇。然后，针对每个簇心 $a_j$，重新计算每个簇的重心，将其定为新的簇心。

$$a_j=\frac{1}{c_i}\sum_{x\in c_i}x \tag{8}$$

其中 $c_i$ 表示簇心 $a_i$ 所在簇的一点。反复迭代改变簇心的位置，直到聚类中心的距离变化小于设定的阈值。最后，计算聚类内部点到聚类中心的距离 $s(c_i,a_i)$，并将距离按照从小到大排序，按比例 $\theta$ 筛选出距离聚类中心最近的点，作为聚类中心密集点，$\theta$ 的取值范围为[0, 0.5]。在本实验中，$\theta$ 设置为 0.2。

对于聚类中心密集点，由于接近中心且较为密集，它们的 LOF 值近似为 1，不是值得关注的离群点，因此将它们的 LOF 值标记为 1，将其余点标记为噪声点，并对噪声点进行进一步的分析处理。最后在剩余噪声点组成的数据集上使用 LOF 算法，通过式(7)计算 LOF 值，找出异常点。

改进的 LOF 算法输入为数据集 $D$，邻居数 $k$，输出异常点 $o$。对于每个噪声数据点，进行 $k$ 个邻居查询，得到 $k$ 个邻居，各数据对象的局部离群因子可由式(7)求得；对计算得到的局部离群因子进行降序排序，前 $n$ 个数据点为异常点。

## 2.4 结果融合

算法的整体流程如图 2 所示。将改进后的 iForest 和 LOF 两个算法输出的结果映射到同一空间并计算得出最终的异常值。判断所考察数据点是否为异常点，需要从其在全局数据空间所处位置以及邻域密度差异两方面来考虑。iForest 根据数据全局信息计算数据点异常分数，LOF 根据数据点邻域信息计算 lof 值，因此，本方法从数据点的全局分布下的异常信息以及局部离群程度来综合确定最终离群点。结合两个算法所得数据点 $a$ 的异常分数 $s_iForest(a)$ 以及 $s_lof(a)$，分别计算它们的 $Z-score$ 值，将它们标准化到同一空间。

$$zs_iforest(a)=\frac{s_iforest(a)-mean_iforest}{std_iforest} \tag{9}$$

$$zs_lof(a)=\frac{s_lof(a)-mean_lof}{std_lof} \tag{10}$$

其中，$mean_iForest$ 、$std_iForest$ 、$mean_lof$ 、$std_lof$ 分别是各自算法结果中所有异常值的均值与标准差。最后，根据式(11)计算得到数据点 $a$ 最终的异常值 $score(a)$ 。其中，$\beta$ 为融合权重，由于 LOF 算法检测更精确，具有更高可信度，且孤立森林算法具有随机性，经过实验验证，赋予 lof 值较大权重，取值范围区间[0.6, 0.8]。

$$score(a)=(1-\beta)*zs_iforest(a)+\beta*zs_lof(a) \tag{11}$$

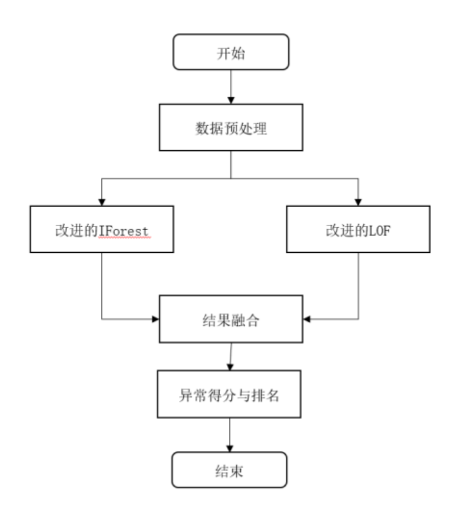



图 2　算法流程图

Fig. 2　Flow chart of algorithm

最后输出所有用户的异常值，并从大到小进行排序。排名越靠前用户越有可能是异常用户。

## 3　实验与结果分析

### 3.1　实验细节

实验中的模型均是用 python 3.6 实现，实验环境为 Windows 10,CPU 为 i7-10700。

### 3.2　实验数据

本文所用到的数据集是从本实验室出口交换机自行收集得到的真实上网流量，包含 120 个上网节点。初始原始数据为 10GB 的 pcap 格式文件，经过流处理程序处理，将每个 pcap 文件中的数据包(packet)以流(flow)为单位进行归并，并提取出 n 维流特征信息：每条流的用户 IP、连接时间、出流量大小、出流量峰值、出流量均值、入流量大小、入流量峰值、入流量均值、出包大小、出包峰值、出包均值、入包大小、入包峰值、入包均值等。本次实验截取了 2021 年 6 月 28 日 24 小时的数据，包含一千万条数据信息。部分流量日志截图如图 3 所示

| time | sIP | dIP | outlen | inlen | outlen1 | inlen1 | maxoutlen | maxinlen |
|---|---|---|---|---|---|---|---|---|
| 2021/6/28 6:51 | 192.168.0. | 172.17.0.1 | 56 | 0 | 15241 | 28618 | 15177 | 28618 |
| 2021/6/28 6:51 | 192.168.0. | 172.17.0.1 | 56 | 0 | 14334 | 13271 | 14334 | 13215 |
| 2021/6/28 6:51 | 192.168.0. | 172.17.0.1 | 56 | 0 | 439 | 511 | 439 | 433 |
| 2021/6/28 6:51 | 192.168.0. | 172.17.0.1 | 56 | 0 | 6881 | 4280 | 6680 | 4280 |
| 2021/6/28 6:51 | 192.168.0. | 172.17.0.1 | 56 | 0 | 3776 | 9195 | 3707 | 9195 |
| 2021/6/28 6:51 | 192.168.0. | 172.17.0.1 | 56 | 0 | 5196 | 8466 | 4966 | 8466 |
| 2021/6/28 6:51 | 192.168.0. | 172.17.0.1 | 56 | 0 | 6513 | 0 | 6461 | 0 |
| 2021/6/28 6:51 | 192.168.0. | 172.17.0.1 | 56 | 0 | 208 | 208 | 208 | 156 |
| 2021/6/28 6:51 | 192.168.0. | 183.236.55 | 0 | 1500 | 104 | 0 | 52 | 0 |
| 2021/6/28 6:51 | 192.168.0. | 183.236.55 | 0 | 974 | 5191 | 2212 | 5191 | 2172 |
| 2021/6/28 6:51 | 192.168.0. | 183.236.55 | 40 | 0 | 6975 | 8138 | 6975 | 8098 |
| 2021/6/28 6:51 | 192.168.0. | 120.241.17 | 0 | 137 | 4430 | 1633 | 4378 | 1633 |
| 2021/6/28 6:51 | 192.168.0. | 183.236.55 | 553 | 0 | 8596 | 8404 | 8431 | 8404 |
| 2021/6/28 6:51 | 192.168.0. | 183.236.55 | 998 | 0 | 2066 | 3043 | 1982 | 3043 |
| 2021/6/28 6:51 | 192.168.0. | 183.236.55 | 1001 | 0 | 626 | 0 | 586 | 0 |
| 2021/6/28 6:51 | 192.168.0. | 121.51.141 | 40 | 0 | 1787 | 1594 | 1691 | 1594 |
| 2021/6/28 6:51 | 192.168.0. | 52.109.76. | 0 | 40 | 9837 | 12740 | 9837 | 12603 |
| 2021/6/28 6:51 | 192.168.0. | 52.109.76. | 40 | 0 | 4359 | 1777 | 4311 | 1777 |

图 3　部分流量日志截图

Fig. 3　Screenshot of partial network flow log

### 3.3　实验结果与分析

由于收集的真实数据集没有标签，所以本算法为无监督算法。将数据集放入模型中，分别对原始 iForest 算法、原始 LOF 算法和本文所提算法进行测试，结果如表 1 所示。

表 1　实验结果

Tab. 1　Experimental result

| 源 IP | 原始 iForest 得分 | 原始 LOF 得分 | 本文融合方法得分 |
|---|---|---|---|
| 192.168.0.236 | 4.3901 | 4.5328 | 4.5043 |
| 192.168.0.31 | -0.4557 | 4.7656 | 3.7213 |
| 192.168.0.247 | 3.9187 | 3.5272 | 3.6055 |
| 192.168.0.193 | 1.9801 | 2.9107 | 2.7246 |
| 192.168.0.222 | 0.6344 | 2.5291 | 2.1502 |
| 192.168.0.158 | 0.9179 | 1.0917 | 1.0569 |
| 192.168.0.42 | 1.7536 | 0.3421 | 0.6244 |
| 192.168.0.88 | 2.8016 | -0.2931 | 0.3258 |
| 192.168.0.196 | 0.5853 | -0.0250 | 0.0971 |
| 192.168.0.181 | 1.3229 | -0.3214 | 0.0075 |
| …… | …… | …… | …… |

从表 1 可以看出，原始 iForest 算法和原始 LOF 算法的结果有较大差别，对于部分相同源 IP 的异常流量检测结果相差较大，比如对于源 IP192.168.0.31，原始 iForest 算法得到的得分为负，这说明该算法认为该用户为异常用户的可能性很低；而原始 LOF 算法计算出的异常得分却很高，认为该用户极有可能是异常用户。这是由于 iForest 是根据数据全局信息计算数据点异常分数，而 LOF 根据数据点邻域信息计算 lof 值。单一的异常检测方法对所有数据采用同一种异常标准，这就导致无法综合考虑数据的全局和局部信息。当大规模多维数据集的正常点与离群点的比例极其不平衡时，采用统一标准的单一异常检测方法无法准确检测出异常点。

通过本方法融合二者结果，既利用了 iForest 算法对全局数据空间处理的优势，也发挥了 LOF 算法对邻域密度处理的优势，结合数据的全局异常信息以及局部离群程度来综合确定异常点，提高了异常数据的检测精度。最终得到的异常分数与异常用户排名更加合理。

将最终异常得分归一化后通过抖动图可视化如图 4 所示。抖动图将每个数据点从原始位置移动一个小的随机数，目的是确保没有两个点完全落在彼此之上。图 4 中每一个点代表一个源 IP 即用户，横轴表示归一化后的异常得分，得分接近 1 表示算法认为该用户有很大几率是异常用户，在图中表现为靠近右侧；而得分接近 0 则表示算法认为该用户有很大几率是正常用户，在图中表现为靠近左侧。

由图 4 可以看出，异常得分较高的点即潜在异常用户分布稀疏，而异常得分较低的正常点聚集在图片的左侧。这说明本方法能够较为明显从流量日志中分离出的异常用户。

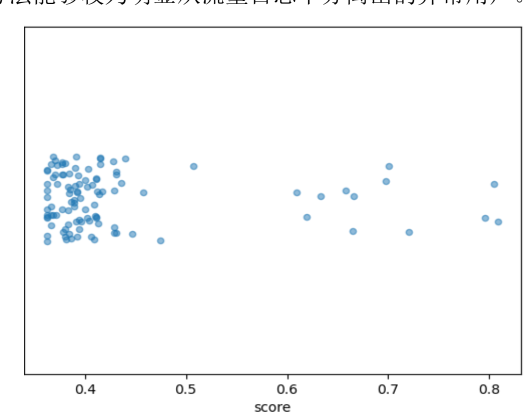



图 4　异常得分抖动图

Fig. 4　Jitter diagram of abnormal score

为了更加精确地衡量算法的精确性，对用数据集中异常源 IP 进行标注，数据集共有 225 个有效用户数据，其中包含 17 个异常用户，异常用户占比为 7.56%。为了体现本文所提算法的优势，除了原始 iForest 与原始 LOF 算法，还同 CBLOF[19]、HBOS[20]、OneClassSVM[21]这三种无监督异常检测算法进行比较。其中，CBLOF 算法是基于聚类的 LOF 算法，通过引入聚类思想分离出异常点,在实验中表现出较好的效果。HBOS 算法是一种单变量方法的组合，对大数据集友好，在全局异常检测问题上表现良好。OneClassSVM 算法有能力捕获数据集的形状，因此对于强非高斯数据有更加优秀的效果。将六种不同的算法运用到标注后的数据集进行测试。图 5 与 6 给出了 6 种不同算法下的 *ROC* 曲线和对应的 *AUC* 值。该曲线的横坐标为假阳性率(False Positive Rate, FPR)，纵坐标为真阳性率(True Positive Rate, TPR)。其中 *AUC* 为 *ROC* 曲线下方的面积，可以通过比较 *AUC* 的大小来评估算法的性能。 *ROC* 曲线一般会位于 $y=x$ 这条线的上方，所以 *AUC* 的值为 0.5~1.0，*AUC* 越接近于 1.0，则说明算法的精确度越高。

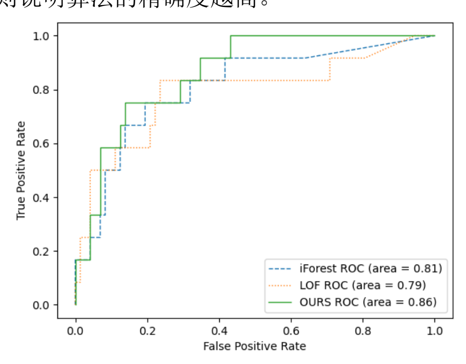



图 5　三种算法的 ROC 曲线图

Fig. 5　ROC curves of trhee algorithms

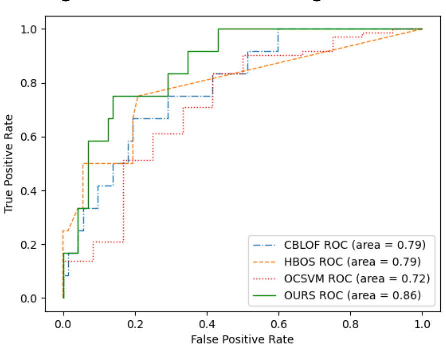



图 6　四种算法的 ROC 曲线图

Fig. 6　ROC curves of four algorithms

从图 5 可以看出，本方法的 *AUC* 值整体高于两个原始算法以及另外三种无监督异常检测算法，而 iForest 算法的 *AUC* 值高于 LOF 算法。在横轴的假阳性率达到 0.4 时，本方法纵轴的真阳性率就已经接近 100%，而原始 iForest 的真阳性率为 90%左右，原始 LOF 仅为 85%左右。从图 6 可以看出，在横轴的假阳性率达到 0.4 时，其余三种无监督异常检测算法的真阳性率则在 70%到 80%之间。通过上述分析可以得出，在相同数据集下，本文提出的算法相比于原始 iForest 和原始 LOF 算法应用于流量异常检测上具有更高的精确度，证明本文所提方法更加适用于流量异常检测分析。

## 4　结束语

在真实网络环境中进行流量异常的检测，往往面临着流量数据规模较大和缺乏有效标注数据这两个挑战性问题。因此研究一种基于无监督学习的快速流量异常检测方法，能够处理大规模的高维流量数据，是本文的研究目标。

针对这一目标，本文提出了一种基于 iForest 和 LOF 的改进型流量异常检测方法。该方法对原始 iForest 和 LOF 算法进行改进并对结果进行融合，在一定程度上弥补了两种算法的缺点并保留了优点，最后通过实验验证了该方法的可行性和有效性。但该方法还存在运行时间较长的问题，因此笔者下一步的研究方向是如何提升算法的运行效率。

## 参考文献：

[1] 李杰铃，张浩. 半监督异常流量检测研究综述 [J]. 小型微型计算机系统, 2020, 41 (11): 2371-2379. (Li Jieling, Zhang Hao. Survey on semi-supervised anomaly traffic detection [J]. Journal of Chinese Computer Systems, 2020, 41 (11): 2371-2379.)

[2] Liu F T, Ting K M, Zhou Z H, “Isolation Forest, ”in Proceedings of the 2008 8th IEEE International Conference on Data Mining. IEEE Computer Society, 2008, pp. 413–422.

[3] Breunig M M, Kriegel H P, Ng R T, *et al.* “LOF: Identifying Density-based Local Outliers, ”in Proceedings of the 2000 ACM SIGMOD International Conference on Management of Data. ACM, 2000, pp. 93–104.

[4] Bay S D, Schwabacher M, “Mining Distance-based Outliers in Near Linear Time with Randomization and a Simple Pruning Rule, ”in Proceedings of the 9th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. ACM, 2003, pp. 29–38.

[5] 王燕晋，易忠林，郑思达，等. 基于孤立森林算法的电力用户数据异常快速识别研究 [J]. 电子设计工程, 2022, 30 (03): 11-14+19. DOI: 10. 14022/j. issn1674-6236. 2022. 03. 003. (Wang Yanjin, Yi Zhonglin, Zheng Sida, *et al.* Research on fast identification of power user data abnormal based on isolation forest algorithm [J]. Electronic Design Engineering, 2022, 30 (03): 11-14+19. DOI: 10. 14022/j. issn1674-6236. 2022. 03. 003.)

[6] 肖峰. 基于孤立森林算法的计算机网络潜在攻击检测方法 [J]. 河北北方学院学报（自然科学版）, 2021, 37 (11): 13-18. (Xiao Feng. Detection method of potential attack on computer network based on Isolated Forest algorithm [J]. Journal of Hebei North University (Natural Science Edition) , 2021, 37 (11): 13-18.)

[7] 徐迪，陆煜锌，肖勇，等. 基于孤立森林算法的配电网线损异常判定 [J]. 电力系统保护与控制, 2021, 49 (16): 12-18. DOI: 10. 19783/j. cnki. pspc. 201267. (Xu di, Lu Yuxin, Xiao Yong, *et al.* Identification of abnormal line loss for a distribution power network based on an isolation forest algorithm [J]. Power System Protection and Control, 2021, 49 (16): 12-18. DOI: 10. 19783/j. cnki. pspc. 201267.)

[8] 杨晓晖，张圣昌. 基于多粒度级联孤立森林算法的异常检测模型 [J]. 通信学报, 2019, 40 (08): 133-142. (Yang Xiaohui, Zhang Shengchang. Anomaly detection model based on multi-grained cascade isolation forest algorithm [J]. Journal on Communications, 2019, 40 (08): 133-142.)

[9] 李倩，韩斌，汪旭祥. 基于模糊孤立森林算法的多维数据异常检测方法 [J]. 计算机与数字工程, 2020, 48 (04): 862-866. (Li Qian, Han Bin, Wang Xuxiang. Multidimensional data anomaly detection method based on fuzzy Isolated Forest algorithm [J]. Computer & Digital Engineering, 2020, 48 (04): 862-866.)

[10] 赵嫚，李英娜，李川，等. 基于模糊聚类和孤立森林的用电数据异常检测 [J]. 陕西理工大学学报（自然科学版）, 2020, 36 (04): 38-43. (Zhao Man, Li Yingna, Li Chuan, *et al.* Anomaly detection of power consumption data based on fuzzy clustering and Isolated Forest [J]. Journal of Shaanxi University of Technology (Natural Science Edition) , 2020, 36 (04): 38-43.)

[11] 李新鹏，高欣，阎博，等. 基于孤立森林算法的电力调度流数据异常检测方法 [J]. 电网技术, 2019, 43 (04): 1447-1456. DOI: 10. 13335/j. 1000-3673. pst. 2018. 0765. (Li Xinpeng, Gao Xin, Yan Bo, *et al.* An approach of data anomaly detection in power dispatching streaming data based on Isolation Forest algorithm [J]. Power System Technology, 2019,

43 (04): 1447-1456. DOI: 10. 13335/j. 1000-3673. pst. 2018. 0765.)

[12] 王巨灏，蔡嘉辉，王琨，等. 基于 KNN 与 LOF 算法的台区线损异常检测 [J]. 电工技术, 2021 (24): 175-177. DOI: 10. 19768/j. cnki. dgjs. 2021. 24. 059. (Wang Juhao, Cai Jiahui, Wang Kun, *et al.* Detection of abnormal line loss in station area based on KNN and LOF algorithm [J]. Electric Engineering, 2021 (24): 175-177. DOI: 10. 19768/j. cnki. dgjs. 2021. 24. 059.)

[13] 刘芳，齐建鹏，于彦伟，等. 基于密度的 Top-n 局部异常点快速检测算法 [J]. 自动化学报, 2019, 45 (9): 1756-1771. (Liu Fang, Qi Jianpeng, Yu Yanwei, *et al.* A fast algorithm for density-based Top-n Local Outlier Detection [J]. ACTA AUTOMATICA SINICA, 2019, 45 (9): 1756-1771.)

[14] 曾冬洲，郑宗华. 基于局部离群因子算法的变压器异常检测 [J]. 电气开关, 2021, 59 (02): 12-15+20. (Zeng Dongzhou, Zheng Zonghua. Transformer anomaly detection based on Local Outlier Factor algorithm [J]. Electric Switchgear, 2021, 59 (02): 12-15+20.)

[15] 司方远，韩英华，赵强，等. 基于 AP-LOF 离群组检测的配电网连接验证 [J]. 东北大学学报（自然科学版）, 2020, 41 (08): 1070-1074. (Si Fangyuan, Han Yinghua, Zhao Qiang, *et al.* Verification of distribution network connectivity based on AP-LOF outlier group detection [J]. Journal of Northeastern University (Natural Science) , 2020, 41 (08): 1070-1074.)

[16] Xu Z, Kakde D, Chaudhuri A. Automatic Hyperparameter Tuning Method for Local Outlier Factor, with Applications to Anomaly Detection [J]. arXiv preprint arXiv: 190200567. 2019.

[17] 仇开，姜瑛. 加权 LOF 结合上下文判断的云环境中服务运行数据异常检测方法 [J]. 计算机工程与科学, 2020, 42 (06): 951-958. (Qiu Kai, Jiang Ying. A service running data anomaly detection method based on weighted LOF and context judgment in cloud environment [J]. Computer Engineering & Science, 2020, 42 (06): 951-958.)

[18] 贺寰烨，林果园，顾浩，等. 云虚拟机异常检测场景下改进的 LOF 算法 [J]. 计算机工程与应用, 2020, 56 (23): 80-86. (He Huanye, Lin Guoyuan, Gu Hao, *et al.* Improved LOF algorithm in cloud virtual machine anomaly detection scenario [J]. Computer Engineering and Applications, 2020, 56 (23): 80-86.)

[19] He Z, Xu X, Deng S. Discovering cluster-based local outliers [J]. Pattern recognition letters, 2003, 24 (9-10): 1641-1650.

[20] Goldstein M, Dengel A. Histogram-based outlier score (hbos): A fast unsupervised anomaly detection algorithm [J]. KI-2012: poster and demo track, 2012, 9.

[21] Chen Y, Zhou X S, Huang T S. One-class SVM for learning in image retrieval [C]// Proceedings 2001 International Conference on Image Processing (Cat. No. 01CH37205) . IEEE, 2001, 1: 34-37.